Progetti
Raspberry Pi
Pico

Raspberry Pi

Official Magazine
#163 | March 2026

La rivista ufficiale Raspberry Pi tradotta in italiano per RaspberryItaly

PICCOLA INTRODUZIONE A

PYTHON

Piccoli script, risultati rapidi

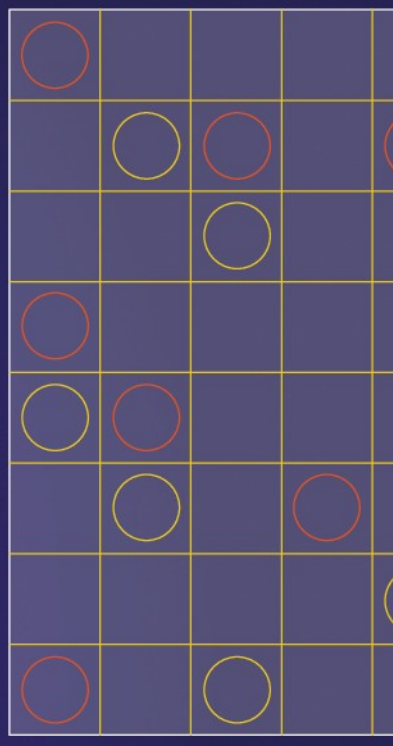
CC BY NC SA

Estratto dal numero 163 di Raspberry Pi Official Magazine. Traduzione di *Zzed* e *marcolecce*, revisione testi e impaginazione di Mauro *"Zzed"* Zoia (zzed@raspberryitaly.com), per la comunità italiana Raspberry Pi www.raspberryitaly.com. Distribuito con licenza CC BY-NC-SA 3.0. *Raspberry Pi Official Magazine* è pubblicato mensilmente da Raspberry Pi Ltd., 194 Cambridge Science Park, Milton Road, Cambridge, England, CB4 0AB.

PICCOLA INTRODUZIONE A PYTHON

Inizia oggi stesso a imparare Python. Scopri le librerie che ti aiutano a creare rapidamente codice potente.

Di **Sean McManus**.



Hai bisogno di un'app semplice? Perché non crearla tu stesso? Utilizzando il linguaggio di programmazione Python, puoi dare precise istruzioni al computer per fare tutto ciò che desideri.

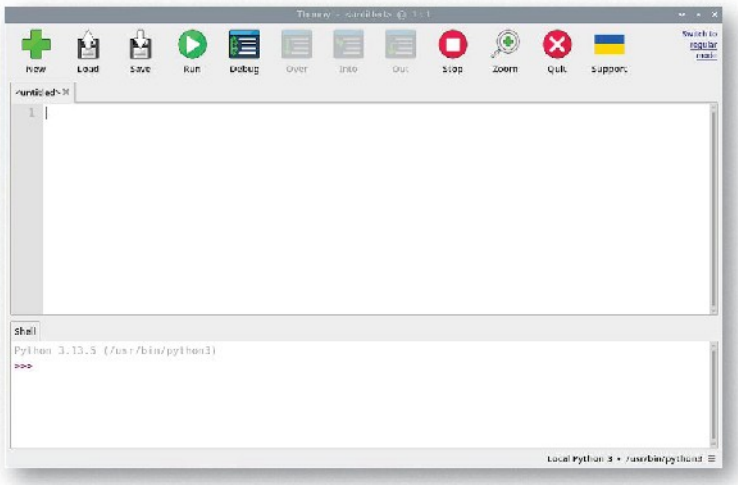
Python viene descritto come "tutto incluso" perché ha molte funzionalità

Python è il linguaggio di programmazione più popolare al mondo, secondo il TIOBE Index (**rpimag.co/tiobe**), ed è facile capirne il perché. I programmi sono chiari e facili da leggere, con molte meno parentesi tra cui districarsi, rispetto ad altri linguaggi.

È anche potente. Python viene spesso descritto come un linguaggio "tutto incluso" perché ha tantissime funzionalità integrate. Puoi estendere il linguaggio aggiungendo librerie per fare cose come scaricare dati, creare codici QR e modificare file PDF. Di conseguenza, puoi fare molto con poche righe di codice.

In questo articolo, vedrai come ottenere rapidamente risultati da Python su Raspberry Pi. Introdurremo alcune delle basi del linguaggio, poi condivideremo alcuni piccoli programmi interessanti. Sperimenta. Divertiti a modificarli. Inizia a programmare oggi!

Scrivi il tuo primo programma Python



▲ Cosa vede ogni programmatore prima di scrivere il suo prossimo grande script?

Il modo migliore per imparare a programmare è farlo, quindi creiamo insieme un semplice gioco a quiz. Non preoccuparti se il codice ti sembra incomprensibile. Ti spiegheremo tutto passo dopo passo.

01 Apri Thonny

Raspberry Pi OS include Thonny, che puoi utilizzare per scrivere ed eseguire programmi Python. Lo trovi nella sezione "Programmazione" del menu del desktop di Raspberry Pi.

02 Digita il tuo primo programma

Thonny ha due pannelli. In basso si trova la "Shell". Se inserisci istruzioni qui, queste vengono eseguite immediatamente. Ci concentreremo sul pannello superiore, dove si digitano i programmi. I programmi sono insiemi di istruzioni memorizzate che è possibile salvare, riutilizzare e modificare facilmente. Inserisci il seguente breve programma. Riesci a prevedere cosa farà?

```
name = input("Come ti chiami?")
print ("Ciao", name)
```

Puoi ottenere risultati interessanti con molto poco codice

In Thonny, noterai che "Come ti chiami?" e "Ciao" sono in verde, il che significa che sono blocchi di testo, noti come "stringhe". Le stringhe sono racchiuse tra virgolette.

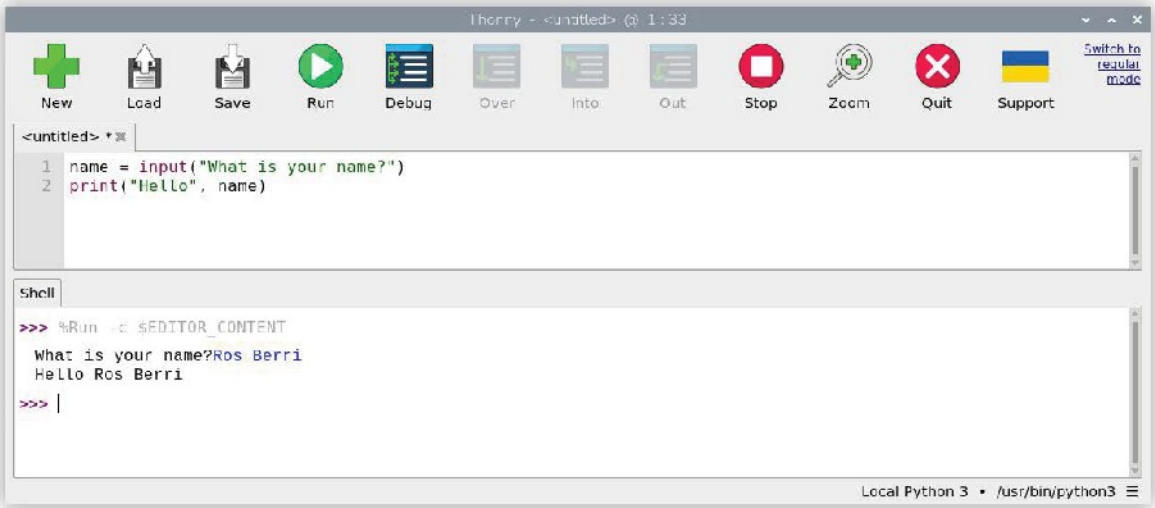
03 Lancia il tuo programma

Fai clic sul pulsante verde Esegui per avviare il programma. Se non funziona, ricontrolla il codice, soprattutto la punteggiatura.

La prima riga utilizza la funzione `input()` di Python, che indica all'utente cosa inserire e attende che lo digiti. L'input dell'utente viene memorizzato in una variabile chiamata `name`. Una variabile è come una scatola per memorizzare le informazioni. Un programma può modificare (o variare) le informazioni contenute in una variabile.

Nella seconda riga, la funzione `print()` visualizza le informazioni nella Shell. All'interno delle parentesi quadre inseriamo le informazioni che vogliamo visualizzare. Si tratta della stringa `"Ciao"` e della variabile che contiene il nome dell'utente. Le separiamo con una virgola. Puoi modificare il programma in modo che visualizzi "[nome] è fantastico"?

▼ Scrivi il tuo programma nel pannello superiore e visualizza l'output nella Shell



04 Aggiungi un quiz a domande

Estendiamo questo programma in un quiz "Indovina l'anno", come mostrato in **simple_quiz.py**.

Il gioco inizia alla riga 5. Poniamo la domanda del quiz e memorizziamo la risposta del giocatore in una variabile chiamata `guess`. La funzione `input()` crea variabili stringa, quindi la riga 6 converte `guess` in un numero intero in modo da poter eseguire confronti numerici. Il simbolo # indica un commento, aggiunto per aiutare le persone a comprendere il codice. Il computer ignora i commenti.

Le righe 7 e 8 vanno di pari passo. La riga 7 controlla se il numero nella variabile `guess` è uguale a 1837, la risposta corretta. Python usa due segni di uguale per verificare se due elementi sono identici. (Nelle righe 1, 5 e 6, hai visto un segno di uguale usato per significare "rendi questa variabile uguale a" qualcosa.) L'istruzione alla riga 8 viene eseguita solo se la risposta del giocatore è 1837.

Puoi capire che la riga 8 dipende dalla riga 7 perché è rientrata di quattro spazi. Python usa l'indentazione per mostrare quali istruzioni appartengono allo stesso blocco, senza bisogno di punteggiatura aggiuntiva come le parentesi. Se sbagli l'indentazione, il programma non verrà eseguito. Se ti ricordi i due punti alla fine della riga 7 quando digiti il programma, Thonny rientrerà automaticamente la riga successiva. Quando non hai più bisogno dell'indentazione, premi il tasto **CANC**.

Analogamente, le righe 9 e 10 formano una coppia che controlla se la risposta del giocatore è inferiore a 1837 e, in tal caso, gli comunica che ha datoo un numero troppo basso. Le righe 11 e 12 riguardano le ipotesi superiori (o successive) al 1837.

Verifica il funzionamento del tuo programma giocando con ipotesi anticipate, tardive e corrette. Congratulazioni! Hai creato il tuo primo gioco in Python!

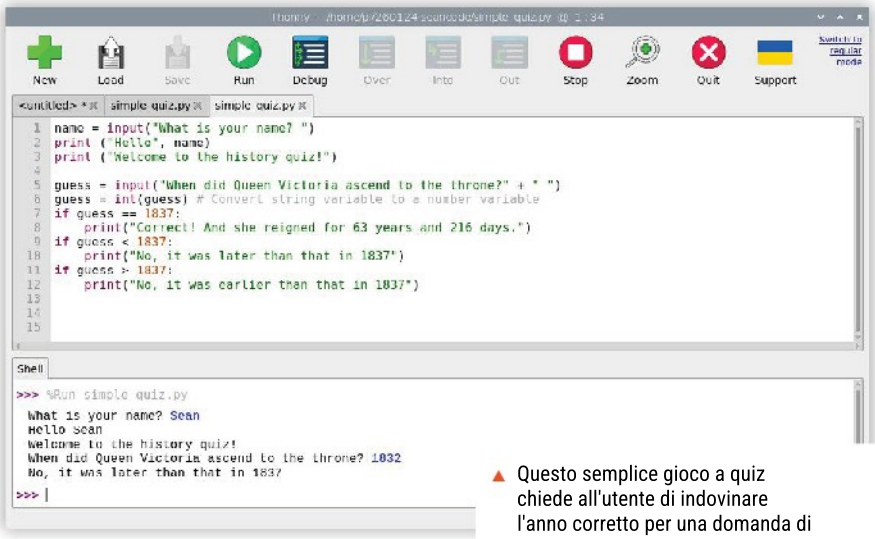
simple_quiz.py

> Linguaggio: **Python**

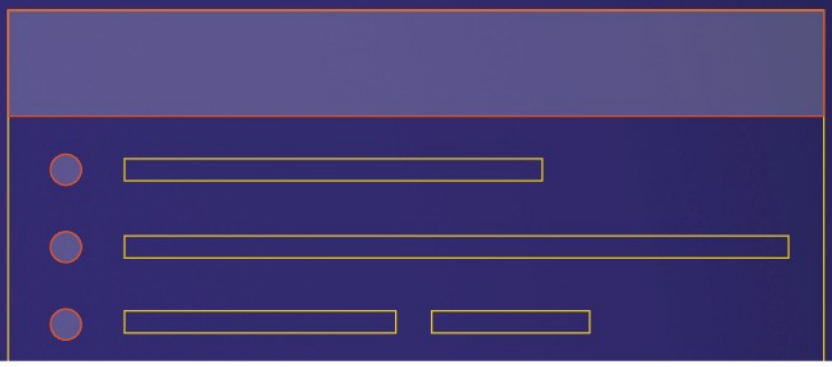
SCARICA IL CODICE COMPLETO:

rpimag.co/python163

```
001.  name = input("Come ti chiami?")
002.  print ("Ciao", name)
003.  print ("Benvenuto al quiz di storia!")
004.
005.  guess = input("Quando salì al trono la Regina
      Vittoria?")
006.  guess = int(guess) # Converti una variabile stringa
      in una variabile numerica
007.  if guess == 1837:
008.      print("Esatto! E regnò per 63 anni e 216
      giorni.")
009.  if guess < 1837:
010.      print("No, fu più tardi, nel 1837.")
011.  if guess > 1837:
012.      print("No, era prima, nel 1837")
```



▲ Questo semplice gioco a quiz chiede all'utente di indovinare l'anno corretto per una domanda di storia.



▼ Questo menu è generato da un elenco di gusti di tè.

```
Shell
>>> %Run tea_menu.py
  Our tea menu:

  * A lovely cup of camomile tea
  * A lovely cup of mint tea
  * A lovely cup of blackcurrent tea
  * A lovely cup of fennel tea

  Our first list item is camomile
  Our second list item is mint
  We have 4 drinks
>>>
```

tea_menu.py

> Linguaggio: **Python**

SCARICA IL CODICE COMPLETO:

rpimag.co/python163

```
001.  print("La nostra carta dei tè:\n")
002.
003.  tea_menu = ["la camomilla", "la menta", " ribes
                 nero", " finocchio"]
004.  for flavour in tea_menu:
005.      print("* Una bella tazza di"+ flavour)
006.
007.  print("\nIl primo della lista è al"+ tea_menu[0])
008.  print("Il secondo della lista è al"+ tea_menu[1])
009.  print("Abbiamo", len(tea_menu), "gusti")
```

05 Capire le liste

Potresti aggiungere altre domande copiando le righe da 5 a 12, incollandole alla fine del programma scritto finora e modificando il codice per ogni domanda. Ma c'è un modo migliore.

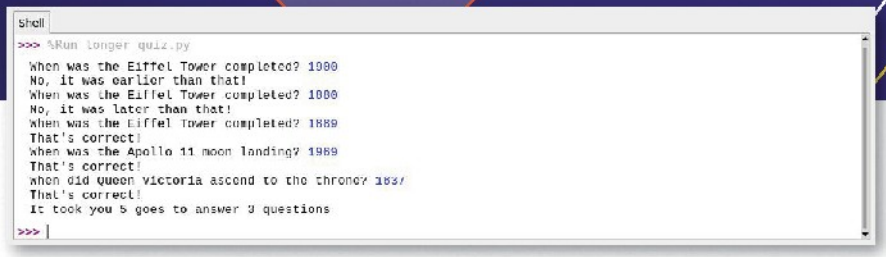
Le liste sono una delle strutture dati che puoi usare per memorizzare informazioni in Python, in modo da poter scrivere programmi più brevi, chiari e sofisticati. Fai clic sul pulsante Nuovo di Thonny per aprire una nuova scheda e inserisci il listato di **tea_menu.py**.

In questo programma, la riga 3 crea una lista chiamata `tea_menu`. Contiene quattro stringhe. Ogni elemento della lista è separato da una virgola e l'intera lista è racchiusa tra parentesi quadre.

Le righe 4 e 5 costituiscono un ciclo `for`. Un ciclo è una porzione di codice che si ripete. La riga 4 imposta il ciclo in modo che ogni volta che il ciclo inizia, il programma prenda l'elemento successivo dalla lista `tea_menu` e lo inserisca in una variabile chiamata `flavour`. La riga 5 è indentata per indicare che appartiene al ciclo. Viene visualizzato il gusto del tè. Prova ad aggiungere altri gusti all'elenco.

Possiamo anche accedere a un elemento dell'elenco utilizzando il suo numero di posizione nell'elenco, chiamato indice. I numeri iniziano da zero, quindi le righe 7 e 8 mostrano come ottenere i primi due gusti. La funzione `len()` rivela la lunghezza di una lista, come mostrato alla riga 9.

C'è molto altro da imparare sulle liste, ma questo è tutto ciò che ci serve sapere oggi. Curioso di sapere cosa significa `\n` nelle stringhe che stampiamo? Aggiunge una riga vuota per riordinare l'output.



▶ Test del gioco a quiz con risposte in anticipo, in ritardo e corrette.

06 Creare un quiz basato su liste

Uniamo tutto ciò che hai visto finora con un paio di nuove idee. Creeremo un gioco a quiz con diverse domande. Il giocatore gioca finché non risponde correttamente, dopodiché il programma passa alla domanda successiva. Consulta il file **longer_quiz.py**.

La prima novità: le liste possono contenere altre liste. Questo programma crea una lista chiamata **question_and_answer_list** (righe da 3 a 7). Ogni elemento è una lista che contiene una singola domanda (una stringa) e la sua risposta (un numero).

La riga 9 imposta un ciclo che scorre la lista di domande e risposte. Estrae la successiva coppia domanda-risposta e la memorizza in due variabili chiamate **question** e **answer**.

La seconda novità è la possibilità di avere cicli in altri cicli, chiamati cicli annidati.

Ti serve una app semplice? Perché non te la fai da solo?

Questo programma introduce il ciclo **while** a partire dalla riga 12 e fino alla riga 21. Questo tipo di ciclo ripete un blocco di codice finché una condizione è vera. In questo caso, ripetiamo finché la risposta del giocatore non è uguale **(!=)** alla risposta corretta. Abbiamo indentato le righe seguenti di altri quattro spazi per indicare quali righe appartengono a questo ciclo. Quindi, abbiamo un ciclo esterno che scorre le domande e un ciclo interno che prende la risposta del giocatore, gli dice se è in anticipo, in ritardo o corretta e ripete finché non indovina.

A quel punto, il programma torna alla riga 9 per prendere la coppia domanda-risposta successiva.

La riga 1 crea una variabile per ricordare quanti tentativi ha fatto il giocatore. Viene incrementata alla riga 15. Alla fine, diamo al giocatore il suo punteggio. Alla riga 13, è stato aggiunto uno spazio alla fine della domanda, per rendere più ordinata la visualizzazione. Altrimenti, la risposta del giocatore si unisce alla domanda.

Personalizzare i programmi è una buona pratica quando si impara Python. Aggiungi le tue domande e modifica i messaggi in questo quiz!

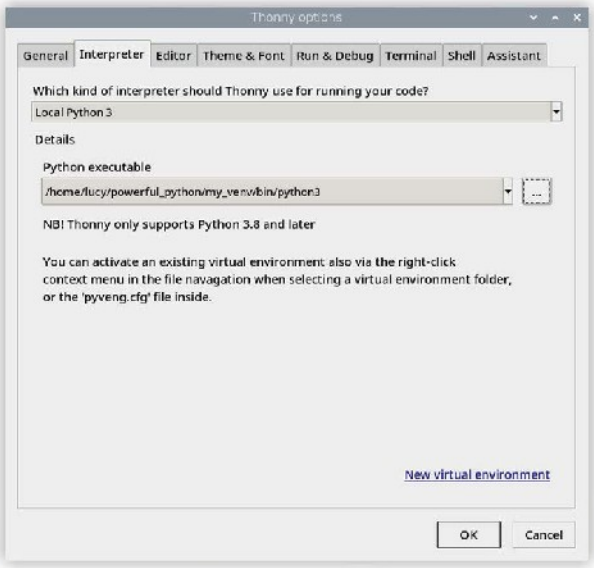
longer_quiz.py

> Linguaggio: **Python**

SCARICA IL CODICE COMPLETO:
rpimag.co/python163

```
001. guess_count = 0
002.
003. question_and_answer_list = [
004.     ["Quando è stata completata la torre Eiffel?",
     1889],
005.     ["Quando allunò Apollo 11?",
     1969],
006.     ["Quando salì al trono la Regina Victoria",
     1837]
007.     ]
008.
009. for question, answer in
     question_and_answer_list:
010.     guess = None
011.
012.     while guess != answer:
013.         guess = input(question + " ")
014.         guess = int(guess)
015.         guess_count +=1 # aumenta di 1 il
     numero di tentativi
016.         if guess == answer:
017.             print("Esatto!")
018.         if guess < answer:
019.             print("No, è stato più tardi
     di così!")
020.         if guess > answer:
021.             print("No, è stato prima di
     così!")
022.
023. print("Ci hai messo", guess_count, "tentativi
     per rispondere a", len(question_and_answer
     _list), "domande")
```

Estendere Python con le librerie



▶ Scegli il tuo ambiente virtuale in Thonny per utilizzare le tue librerie Python preferite

È **possibile installare facilmente delle librerie aggiuntive che offrono nuove funzionalità con cui sperimentare.** Per farlo, è necessario utilizzare un ambiente virtuale. Si tratta di un ambiente sicuro in cui installare le librerie senza interferire con la versione principale di Python utilizzata dal sistema operativo. Segui questi passaggi.

01 Creare un ambiente virtuale

Utilizza il pulsante in alto a sinistra del desktop per aprire una finestra del Terminale. Crea una cartella per i tuoi progetti chiamata **powerful_python**:

```
$ mkdir powerful_python
$ cd powerful_python
```

Ora crea l'ambiente virtuale:

```
$ python3 -m venv my_venv
```

Questo crea un ambiente virtuale chiamato `my_venv`.

02 Attiva il tuo ambiente virtuale

Il passo successivo consiste nell'attivare l'ambiente virtuale:

```
$ source my_venv/bin/activate
```

Ora dovresti vedere il nome del tuo ambiente virtuale prima del prompt.

03 Installa i pacchetti

Per installare le librerie, utilizziamo il gestore di pacchetti Python, chiamato pip. Ad esempio, per installare la libreria matplotlib, utilizziamo:

```
$ pip install matplotlib
```

04 Usa il tuo ambiente virtuale

Hai creato il tuo ambiente virtuale e installato un pacchetto al suo interno, ma Thonny non lo sta ancora utilizzando.

Fai clic sul menu con le tre linee in basso a destra di Thonny e seleziona "Configura interprete". Ora, fai clic sul pulsante con i tre puntini per aprire la finestra File. Seleziona Home e vai alla cartella **powerful_python**, poi a **my_venv** e **bin** (abbreviazione di binary, non di spazzatura). Seleziona **python3** e fai clic su OK. Questo garantirà che Thonny utilizzi i pacchetti Python presenti nel tuo ambiente virtuale.

Problemi comuni

Senza un ordine particolare, ecco i principali problemi che potresti incontrare:

1. **Parentesi errate o mancanti.** Python utilizza parentesi quadre, curve e graffe per diversi scopi. Ogni parentesi necessita di una parentesi corrispondente.

2. **Due punti mancanti.** Un'istruzione `for`, `while`, `if`, o `def` necessita di due punti alla fine della riga.

3. **Virgole mancanti.** Gli elementi di una lista devono essere separati da una virgola.

4. **Indentazione errata.** Puoi distanziare il codice all'interno delle liste come preferisci, ma il rientro prima delle istruzioni deve seguire regole precise.

Programmi d'impatto

Potresti non capire ogni riga, ma questi programmi Python mostrano come ottenere risultati interessanti con pochissimo codice. Leggi i listati e le relative descrizioni e provali. Abbiamo mantenuto i programmi il più semplice possibile, ma ci sono alcune funzionalità aggiuntive nel codice online. Puoi scaricare tutti gli esempi all'indirizzo **rpimag.co/python163**

Sense HAT brillante!

Anche se non possiedi un Sense HAT, questo è un ottimo programma da leggere. Imposta ciascun LED su un colore casuale. Ti illumina (hai capito il gioco di parole?) su come è possibile inserire dei loop uno dentro l'altro.

Le prime righe importano le librerie che vogliamo utilizzare, tutte già installate.

Abbiamo un ciclo `while True` che si ripete finché non si interrompe il programma. Al suo interno c'è un ciclo che scorre le righe (il ciclo y). E al suo interno, c'è un ciclo che accende ogni LED nella riga corrente (il ciclo x). Come prima, ogni livello di rientro indica che le istruzioni appartengono al ciclo definito sopra.

In precedenza abbiamo utilizzato un ciclo `for` per scorrere una lista. Qui, utilizziamo la funzione `range()` per creare un elenco di numeri e poi lo scorriamo con un ciclo. In questo modo, le variabili y e x assumono i valori da 0 a 8, in sequenza, corrispondenti alle coordinate dei LED.

Generiamo numeri casuali tra 0 e 255 utilizzando `random.randint(0, 255)`. Questi numeri determinano la quantità di rosso, verde e blu nel colore del LED.

Il modulo time può essere utilizzato per aggiungere un piccolo ritardo, in questo caso un quarto di secondo (riga 12).

sparkle.py

> Linguaggio: **Python**

SCARICA IL CODICE COMPLETO:

rpimag.co/python163

```
001. from sense_hat import SenseHat
002. import random
003. import time
004.
005. sense = SenseHat()
006. sense.show_message("Brilla!")
007.
008. while True:
009.     for y in range(8):
010.         for x in range(8):
011.             sense.set_pixel(x, y, [random.
     randint(0, 255), random.randint(0, 255),
     random.randint(0, 255)])
012.             time.sleep(0.25)
```

▲ Utilizzando cicli annidati e numeri casuali, facciamo lampeggiare i LED

Fare arte con il codice

Questo programma disegna quattro stelle sullo schermo, utilizzando il modulo turtle, incluso con Python.

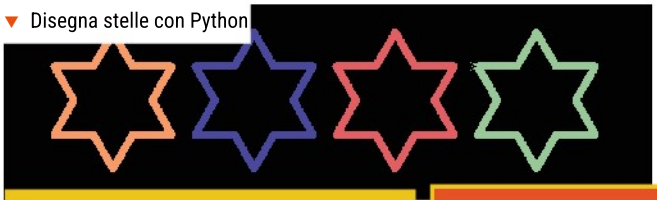
Usiamo un ciclo **for** con **range()** da ripetere un numero prestabilito di volte, ma impostiamo la variabile col carattere di sottolineatura. Significa che non è necessario utilizzare il numero nel range.

Il ciclo che inizia alla riga 16 disegna una stella con sei punte. Si trova all'interno del ciclo che inizia alla riga 10, il quale si ripete quattro volte per disegnare quattro stelle.

Nella riga 14, il colore della penna è composto da tre numeri casuali, nello stesso modo in cui abbiamo assegnato colori casuali ai LED nel programma Sense HAT.

Puoi ora disegnare delle stelle in posizioni casuali?

▼ Disegna stelle con Python



turtle_stars.py

SCARICA IL CODICE COMPLETO:
rpimag.co/python163

> Linguaggio: **Python**

```
001.  import turtle
002.  import random
003.
004.  turtle.colormode(255)
005.  turtle.bgcolor("black") # colore di sfondo
006.  turtle.pensize(10)
007.  turtle.penup()
008.  turtle.goto(-525, 0) # coordinate x, y
009.
010.  for _ in range(4): # Ciclo per fare 4 stelle
011.      turtle.penup()
012.      turtle.forward(180) # spazio tra le stelle
013.      turtle.pendown()
014.      turtle.pencolor(random.randint(0, 255),
      random.randint(0, 255), random.randint(0, 255))
015.
016.      for _ in range(6): # ciclo disegna stella
017.          turtle.forward(50)
018.          turtle.left(60)
019.          turtle.forward(50)
020.          turtle.right(120)
```

▶ Segno crea codici QR

Genera codici QR colorati

La libreria Segno permette di creare codici QR colorati. Utilizza codici colore in esadecimale, un sistema di numerazione che somma le lettere dalla A alla F per rappresentare i numeri da 10 a 16. È possibile trovare online schemi di colori con codici esadecimali. Aggiungi il tuo link alla riga 2. Per ulteriori funzionalità, incluso il supporto per codici Wi-Fi e sfondi animati, consulta la documentazione su **rpimag.co/segnodocs**. Installa la libreria Segno nel tuo ambiente virtuale utilizzando pip install segno.

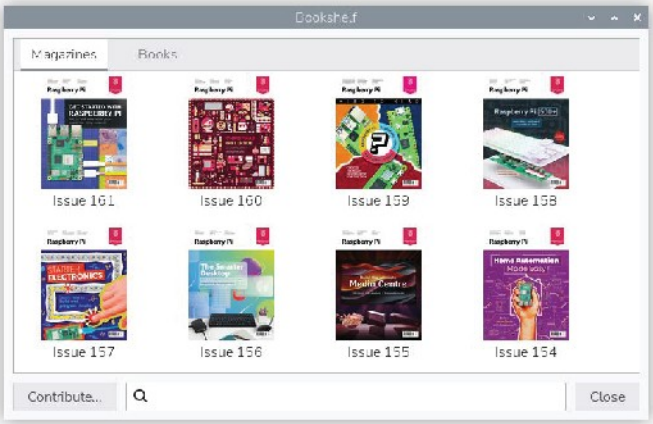
qr_code_maker.py

SCARICA IL CODICE COMPLETO:
rpimag.co/python163

> Linguaggio: **Python**

```
001.  import segno
002.  url = "https://www.raspberryitaly.com"
003.  qr_code = segno.make(url)
004.  qr_code.save("qr_code.png", scale = 12, border = 2, dark =
      "#a600ff", light = "#ffd900", alignment_dark = "#ff007b",
      finder_dark = "#0014f1")
```

Trova i PDF in Bookshelf



Unire i PDF

Quando facciamo ricerche su un argomento, a volte raccogliamo articoli e capitoli di libri pertinenti in un unico file PDF per poterli cercare, stampare o leggere sull'iPad. Questo semplice programma combina le pagine selezionate da diversi file PDF in un unico file. Ogni elemento nella lista dei file è a sua volta una lista che contiene il nome e il percorso del file in una stringa, oltre alle pagine desiderate in un'altra lista. Quindi, abbiamo ua lista dentro una lista dentro una lista. Presta attenzione alle virgole e alle parentesi!

L'app Bookshelf in Raspberry Pi OS fornisce i numeri arretrati di questa rivista. Questa lista di file combina due tutorial Python tratti da numeri precedenti che abbiamo scaricato, e che sono un ottimo punto di partenza dopo aver letto questo articolo. Installa la libreria PyPDF2 nel tuo ambiente virtuale: **`pip install pypdf2`**. Per maggiori dettagli sul funzionamento di PyPDF2 e del salvataggio dei file, consulta l'articolo "Productive Python" nel numero 144 (**rpimag.co/144**).

pdf_merger.py

> Linguaggio: **Python**

SCARICA IL CODICE COMPLETO:

rpimag.co/python163

```
001. from PyPDF2 import PdfReader, PdfWriter
002.
003. file_list = [ ["/home/pi/Bookshelf/MagPi128.pdf", [0, 43,
     44, 45, 46, 47, 48, 49] ],
004.               ["/home/pi/Bookshelf/MagPi82.pdf", [0, 30,
     31, 32, 33, 34, 35, 36] ]
005.               ]
006.
007. w = PdfWriter()
008.
009. for file, page_list in file_list:
010.     r = PdfReader(file)
011.     for p in page_list:
012.         w.add_page(r.pages[p])
013.
014. with open("merged_tutorials.pdf", "wb") as f:
015.     w.write(f)
```

Consulta la tua intensità di carbonio

Il modulo requests consente di scaricare informazioni strutturate da Internet. Questo programma visualizza l'intensità di carbonio attuale per la tua zona di fornitura elettrica (in UK), inclusa la quantità di energia prodotta da carbone ed energia eolica. Un progetto con Raspberry Pi ad alto consumo energetico potrebbe utilizzare queste informazioni per decidere quando eseguire le operazioni che consumano più energia. In alternativa, puoi usarle per decidere quando avviare la lavatrice.

La funzione `split()` di Python trasforma una stringa in una lista. Alla riga 8, divide il codice postale in corrispondenza di ogni spazio e poi prende solo il primo elemento della lista. È un modo elegante per assicurarsi di ottenere solo la prima parte del codice postale. Questo è ciò che dobbiamo inviare all'API (application programming interface) di Carbon Intensity per ottenere i dati corretti.

Questo programma introduce le f-string (vedi righe 9, 16, 17 e 22), che vengono utilizzate per inserire il contenuto di una variabile in una stringa. Si inserisce la `f` prima delle virgolette di apertura e poi si inserisce il nome della variabile tra parentesi graffe all'interno della stringa.

Estrarre frammenti di dati è complicato (vedi righe 11-14). Immagina di navigare lungo un percorso, dove indirizzi il programma lungo i percorsi corretti utilizzando i nomi sui cartelli, combinati con i numeri di indice. Il lavoro più difficile è stato già fatto, quindi puoi semplicemente usare le variabili che abbiamo impostato nel tuo programma. Se sei curioso, puoi vedere un esempio della struttura dei dati su **rpimag.co/carbonintensity**.

Installa le librerie richieste con `pip install requests`.

carbon_intensity.py

SCARICA IL CODICE COMPLETO:

rpimag.co/python163

> Linguaggio: **Python**

```
001. # Carbon intensity reporter
002. # vedi documentazione su
003. # https://carbon-intensity.github.io/api-definitions/#get-
     regional-postcode-postcode
004.
005. import requests
006.
007. postcode = input("Inserisci un codice postale UK: ")
008. postcode = postcode.split(" ")[0]
009. url = f"https://api.carbonintensity.org.uk/regional/postcode/
     {postcode}"
010.
011. received_data = requests.get(url).json()["data"][0]
012. region = received_data["shortname"]
013. intensity = received_data["data"][0]["intensity"]["forecast"]
014. index = received_data["data"][0]["intensity"]["index"]
015.
016. print(f"Regione: {region}")
017. print(f"Intensità di carbonio: {intensity} gCO2/kWh ({index})")
018.
019. for fuel_and_perc in received_data["data"][0]["generationmix"]:
020.     fuel = fuel_and_perc["fuel"]
021.     perc = fuel_and_perc["perc"]
022.     print(f"{fuel} {perc} %")
```

```
Shell
>>> %Run carbon_requests.py
  Enter a UK postcode: CB4
  Region: East England
  Carbon intensity: 143 gCO2/kWh (moderate)
  biomass 5.5 %
  coal 0 %
  imports 2.8 %
  gas 31.1 %
  nuclear 16.8 %
  other 0 %
  hydro 0 %
  solar 3.1 %
  wind 40.6 %
>>>
```

◀ Scopri come viene generata l'energia nella tua zona (in UK), in questo momento.

Utilizza la libreria Pillow per aggiungere filigrane alle tue foto.

Python è il linguaggio di programmazione più popolare al mondo.

Immagini con filigrana

Questo programma applica una filigrana a una immagine e introduce il concetto di funzione, che rappresenta una parte riutilizzabile del programma. La funzione `watermark` è definita alla riga 3 e riceve il nome del file e la sua estensione. Come ci si potrebbe aspettare, le righe rientrate sottostanti appartengono alla funzione. Alla riga 8, si può notare come le stringhe vengano combinate utilizzando il simbolo `+`. La funzione non inizia a funzionare finché la riga 23 non la attiva e le passa le informazioni sull'immagine.

La funzione utilizza la libreria Pillow per aprire il file immagine, creare una sovrapposizione trasparente, disegnare il testo della filigrana su di essa e quindi combinarla con l'immagine originale. L'immagine viene convertita in RGB senza trasparenza in modo da poter essere salvata come JPG. La versione estesa di questo programma online elabora un'intera cartella di immagini e le ridimensiona anche. Installa la libreria utilizzando `pip install pillow`.

watermark_image.py

> Linguaggio: **Python**

SCARICA IL CODICE COMPLETO:

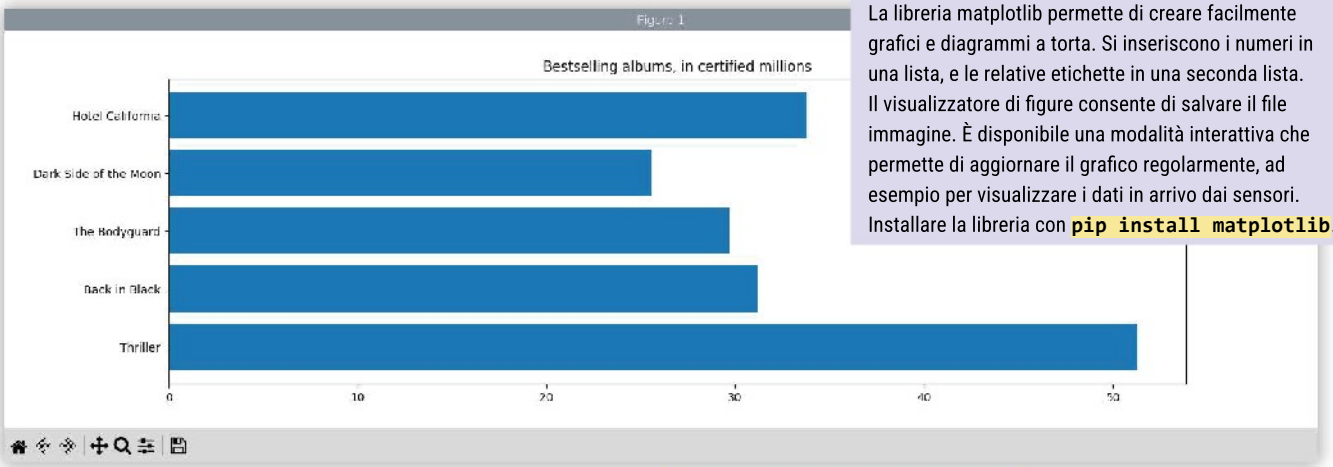
rpimag.co/python163

```
001. from PIL import Image, ImageDraw, ImageFont
002.
003. def watermark(filename, extension):
004.     transparency = 200  # 0 to 255
005.     text = 'Photo by Sean'
006.     font = ImageFont.truetype('DejaVuSans-Bold.ttf', 48)
007.
008.     original = Image.open(filename + "." + extension)
009.     image_height = original.size[1]
010.
011.     text_position = (30, image_height - 70) # Bottom left corner
012.     overlay = Image.new('RGBA', original.size, (255, 255, 255, 0))
013.     draw = ImageDraw.Draw(overlay)
014.     draw.text(text_position, text, font=font, fill=(255, 255,
    255, transparency))
015.
016.     watermarked_image = Image.alpha_composite(
    original.convert('RGBA'), overlay)
017.     watermarked_image = watermarked_image.convert('RGB')
018.
019.     new_filename = filename + "-wm." + extension
020.     watermarked_image.save(new_filename)
021.
022. print("Sto mettendo la filigrana alle tue immagini...")
023. watermark("IMG_3412", "jpg")
```

▼ Genera grafici in Python con la libreria matplotlib

Crea grafici

La libreria matplotlib permette di creare facilmente grafici e diagrammi a torta. Si inseriscono i numeri in una lista, e le relative etichette in una seconda lista. Il visualizzatore di figure consente di salvare il file immagine. È disponibile una modalità interattiva che permette di aggiornare il grafico regolarmente, ad esempio per visualizzare i dati in arrivo dai sensori. Installare la libreria con `pip install matplotlib`.



bar_graph.py

> Linguaggio: **Python**

SCARICA IL CODICE COMPLETO:
rpimag.co/python163

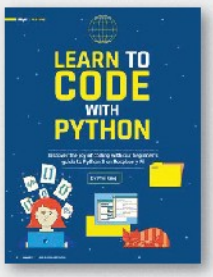
```
001. import matplotlib.pyplot as plt
002.
003. labels = ["Thriller", "Back in Black", "The
     Bodyguard", "Dark Side of the Moon", "Hotel
     California"]
004. values = [51.3, 31.2, 29.7, 25.6, 33.8]
005.
006. plt.barh(labels, values)
007. plt.title("Album più venduti, in milioni di
     copie certificate")
008. plt.show()
```

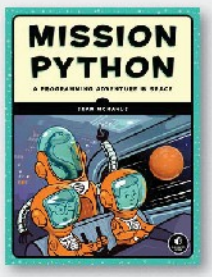
Ulteriori letture

Learn to Code with Python: Ora abbiamo fatto un breve tour per farti ottenere risultati velocemente, ma il nostro articolo del numero 128 copre tutta la sintassi di base di Python. **bit.ly/MagPi128It**

Make Games with Python: Questo libro, Raspberry Pi Essentials, ti mostra come creare giochi con la libreria Pygame. **rpimag.co/makegamesbook**

Mission Python: Crea un gioco 3D ambientato in una stazione spaziale e impara a conoscere le strutture dati e l'animazione lungo il percorso… **rpimag.co/missionpython**





◀ Fai i prossimi passi con Python grazie a questi tutorial.

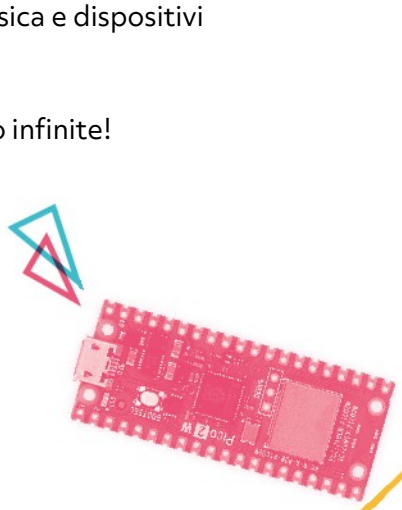
Progetti Raspberry Pi Pico

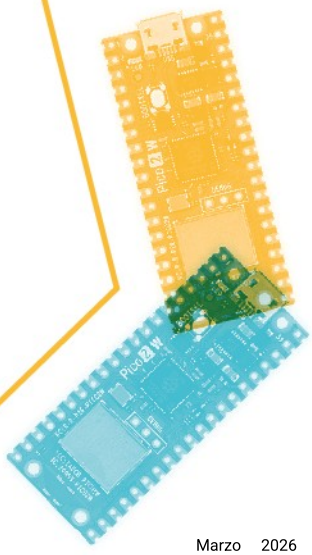
Se hai necessità di una scheda microcontrollore piccola ed economica, con un consumo energetico bassissimo, da integrare in un progetto, Raspberry Pi Pico è l'ideale. È disponibile in diverse versioni, a seconda della potenza di elaborazione necessaria e della necessità di connettività wireless. Tutti i modelli offrono un paio di funzionalità aggiuntive: ingressi analogici e macchine a stati PIO (Programmable Input/Output) che possono gestire alcune attività in background.

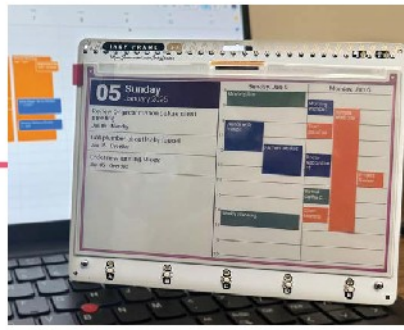
Qui presentiamo solo alcuni dei migliori progetti per Pico per ispirarti, che coprono una vasta gamma di argomenti: da dashboard informative, sensori IoT, illuminazione a LED a robot, droni, musica e dispositivi indossabili.

Le possibilità con un Pico sono davvero infinite!

Di Phil King







Dashboard Informativa

Recupera i dati online e visualizzali su un display

Simple Weather Dashboard

rpimag.co/picosimpleweather

Grazie alla connettività wireless, Pico W (o 2 W) può recuperare informazioni dal World Wide Web, come ad esempio il meteo attuale. Una dashboard meteo è un progetto Pico molto diffuso e il semplice esempio di Pete Cornish è facile da replicare. Tutto ciò che serve è un Pico W e un piccolo schermo per visualizzare le informazioni: lui ha utilizzato un display e-ink Waveshare da 2,13 pollici.

Per connettere Pico W al web, è necessario aggiungere il nome e la password del router wireless a un file di configurazione. In questo esempio, il file contiene anche la latitudine e la longitudine per la posizione e visualizzare le informazioni meteo locali. Per recuperare i dati, è generalmente necessario ottenere una chiave API per il database online che si sta utilizzando, in questo caso OpenWeatherMap.

Inky Dashboard

rpimag.co/inkydashboard

Come mostrato nel numero 152 (**bit.ly/RPOM152it**), l'Inky Dashboard di Jaeheon Shim presenta un calendario e una lista di cose da fare. Funziona su un display e-ink a colori Inky Frame 7.3 con tecnologia Pico. Poiché il layout del calendario utilizza la libreria grafica LVGL, modificare il codice per renderlo adatto a altri display e-ink non dovrebbe essere troppo difficile. I dati del calendario vengono recuperati da iCal, ma è possibile utilizzarli anche con altri servizi come Google Calendar o Microsoft Outlook. Dato che Pico W non è in grado di gestire direttamente questi dati, li recupera tramite un server in esecuzione su un computer o nel cloud. Poiché Pico W recupera i dati solo ogni 30 minuti circa, entrando in una fase di ibernazione profonda tra un recupero e l'altro, si tratta di un progetto molto efficiente dal punto di vista energetico.

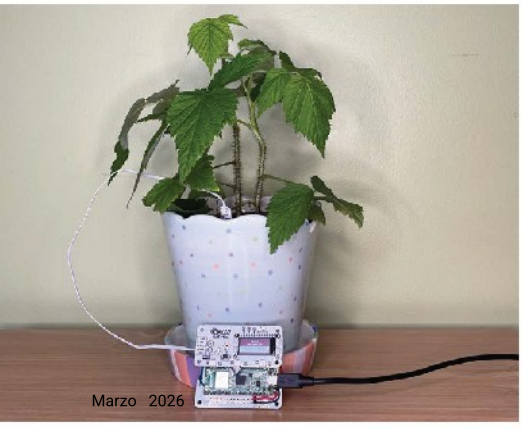
Sensore IoT

Utilizzare Pico come dispositivo IoT per i dati da sensori

Pianta in vaso manda SMS

rpimag.co/picoplant

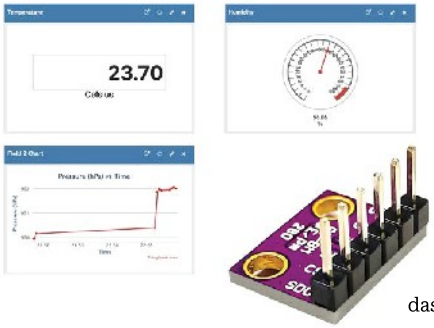
Il progetto di Sandeep Mistry combina un Pico W con un Grow Kit di Pimoroni per monitorare il livello di umidità del terreno. Se è troppo secco, viene inviata una notifica via SMS – tramite l'API di Twilio con un account gratuito – al proprietario per ricordargli di annaffiare. Per un tocco di personalità in più, Sandeep suggerisce di aggiungere messaggi casuali, insieme a un sensore di luce per rilevare l'alba/il tramonto e augurare "buongiorno/buonanotte" di conseguenza. In alternativa, si potrebbe configurare un sistema di auto-irrigazione basato su Pico come quello creato da Veeb: **rpimag.co/sploosh**. Se il sensore rileva terreno secco, attiva un relè per azionare una pompa per acquario che spruzza l'acqua necessaria nel vaso.

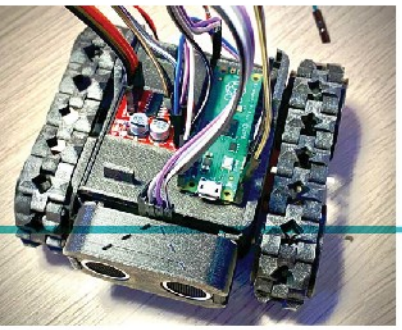


Dashboard IoT

rpimag.co/picothingspeak

Con uno o più sensori collegati a un Pico W, è possibile monitorare l'ambiente locale per aspetti come temperatura, pressione atmosferica e umidità. I dati raccolti possono quindi essere utilizzati a piacimento: in questo progetto di Mahmood M Shilleh, vengono inviati a una dashboard IoT nel cloud, accessibile da qualsiasi dispositivo. Esistono numerosi servizi e metodi utilizzabili a questo scopo; Mahmood ha optato per ThingSpeak, una popolare piattaforma IoT open source. Nella sua guida, mostra come configurare Pico W per inviare dati da un sensore BME280 a ThingSpeak generando una chiave API e creando canali per i flussi di dati, in modo da visualizzarli su una dashboard web.





Robot e Droni

Crea meraviglie meccaniche con Pico

PicoSMARS

rpimag.co/picosmars

Le dimensioni ridotte di Pico consentono di utilizzarlo in robot molto più piccoli rispetto a un normale Raspberry Pi. Per dimostrarlo, RoboticBits ne ha persino realizzato uno con una scheda Pico ridotta e dotata di perline al posto delle ruote: **rpimag.co/tinypicorobot**.

Non devono però essere necessariamente piccoli. Il robot PicoSMARS di Kevin McAleer è un'auto robotica di dimensioni più standard con Pico come cervello. "SMARS" è l'acronimo di "Screwless Modular Assemblable Robotic System" (Sistema Robotico Assemblabile Modulare Senza Viti), un robot stampabile in 3D, modulare e personalizzabile. Pico è collegato a una scheda per azionare i motori, e a un sensore di distanza a ultrasuoni per rilevare gli ostacoli. Il codice e i file STL per la stampa 3D sono disponibili tramite il link.

Drone Quadricottero

rpimag.co/picoquadcopter

Ancora una volta, le dimensioni e il peso ridotti di Pico (4 g, senza connettori) lo rendono perfetto per l'utilizzo nei droni. Creato da Tim Hanewich, questo impressionante drone quadricottero utilizza un Pico come controller di volo: per leggere i dati di telemetria da un accelerometro e giroscopio MPU-6050, interpretare i comandi radio da un ricevitore di bordo e controllare quattro motori indipendenti tramite un ESC (regolatore elettronico di velocità) utilizzando la modulazione di larghezza di impulso (PWM). Il progetto potrebbe sembrare un po' scoraggiante da replicare, ma Tim ha scritto una guida gratuita e open-source in dodici parti sul processo di costruzione, la programmazione in MicroPython e i test per ottenere un volo stabile.

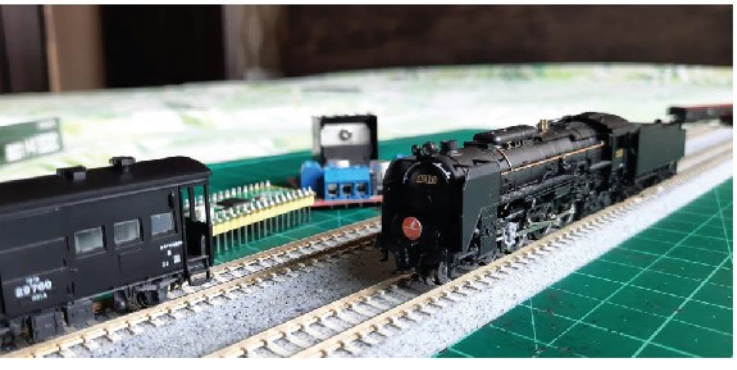
Modellismo ferroviario

Scegli Pico per rimanere sul binario giusto.

Ferrovia automatizzata

rpimag.co/picorailroad

Le schede Raspberry Pi sono state usate da molti appassionati nei loro plastici ferroviari, ad esempio per un'illuminazione smart e controllabile. Questo progetto presenta un "binario con sensori" con sensori di prossimità a infrarossi che rilevano il passaggio di un treno. Pico funge da cervello, leggendo i sensori e controllando la tensione del binario – tramite una scheda di controllo del motore – per modificare la velocità di una locomotiva utilizzando la modulazione di larghezza di impulso (PWM). In questo modo, può farla accelerare, rallentare o arrestare.



Luci Passaggio A Livello

rpimag.co/picolevelcrossing

Collegando Pico a una serie di minuscoli semafori per passaggi a livello di un plastico ferroviario, questo progetto di Pater Practicus (alias Brendan McGrath) rileva il passaggio dei treni e determina quando è sicuro per le macchinine o i pedoni attraversare. In questo modo, combina due degli usi più diffusi di Pico: la capacità di controllare dei LED e di rilevare i dati da un sensore. La versione originale, ancora in fase di sviluppo su un binario di legno, è stata aggiornata a una configurazione più rifinita per un plastico ferroviario motorizzato, pur mantenendo un costo di sole 23 sterline (26 Euro). Il processo di costruzione, inclusi il cablaggio, la programmazione in MicroPython e la realizzazione dei segnali luminosi, è illustrato nel video di YouTube collegato.

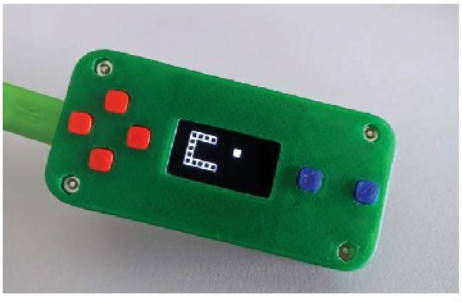
Gaming

Pico è ideale per console palmari

Tiny Game Console

rpimag.co/tinygameconsolevid

Il maker Twan37 ha impiegato solo pochi giorni per realizzare questa console di gioco con Raspberry Pi Pico. Si tratta più che altro di un prototipo, privo di batteria integrata, ma comunque impressionante e dimostra che tipo di console portatile si può facilmente creare con Pico. Alloggiata in un case stampato in 3D, Pico è collegato a un display OLED monocromatico I2C da 0,96 pollici e a pochi pulsanti per i controlli, quattro dei quali per un D-pad improvvisato. Per dimostrare la funzionalità della console compatta, il maker ha creato un paio di semplici giochi in MicroPython, cloni di Snake e Tetris, insieme a un generatore di labirinti.



PicoZX Handheld

rpimag.co/picozxbuild

Un progetto decisamente più sofisticato, questa console portatile Sinclair ZX che emula lo Spectrum si basa su diverse schede PCB personalizzate, con un Pico saldato sulla scheda principale. Ispirato all'emulatore originale PicoZX Spectrum di Peter Misenko, il maker Ken St Cyr (del canale YouTube What's Ken Making) ha persino creato una tastiera QWERTY in miniatura posizionata sotto un display IPS a colori da 2,8 pollici. È presente anche un D-pad integrato, sebbene possa essere utilizzato anche con un joystick separato. Il video di Ken su YouTube documenta l'intero processo di costruzione, se avete voglia di provarci.

Musica

Crea grande musica con Pico

PicoStepSeq

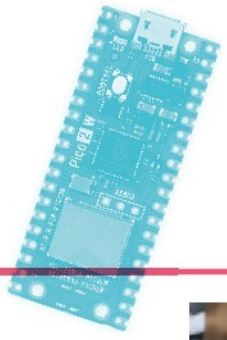
rpimag.co/picostepseq

Sebbene Pico non disponga di un'uscita audio integrata, può essere molto utile per progetti musicali. Presentato nel numero 123 (**rpimag.co/123**), questo sequencer MIDI compatto a otto step è dotato di otto interruttori a leva (con LED di stato), un mini schermo OLED I2C, un encoder rotativo e Pico in un case stampato in 3D. Il maker Tod Kurt afferma che è stato progettato come un potenziale kit fai-da-te per persone con competenze di saldatura di livello base: ad eccezione delle due porte MIDI, tutti i componenti sono di tipo a foro passante. Tutti i dettagli e il codice sono disponibili nel repository GitHub: **rpimag.co/picostepseqgh**.

Controller gestuale MIDI

rpimag.co/midigesture

Un altro progetto musicale MIDI che utilizza Pico, il pedale musicale di GaryRigg per chitarra (o altro strumento), ruota e si muove attorno a un giunto sferico, la cui posizione viene letta da un sensore IMU AHRS a sei assi. A differenza di un pedale standard, qui si possono controllare tre parametri allo stesso tempo con imbardata, beccheggio e rollio, consentendo così di produrre molte più variazioni di effetti. Può anche funzionare manualmente, quindi potrebbe essere utilizzato da DJ o in studio. Per maggiori dettagli, consulta l'articolo di presentazione nel numero 149: **rpimag.co/149**.



Dispositivi Di Input

Realizza un controller con Pico

Gamepad 2.0

rpimag.co/gamepad2

I microcontrollori come Pico sono ideali per alimentare un gamepad o altri dispositivi di controllo. Potresti provare a costruirne uno da zero, ma questo kit di Kevin McAleer lo rende molto più semplice ed è ideale per controllare robot tramite Bluetooth o Wi-Fi.

Il PCB personalizzato permette di saldare semplicemente un Pico W o 2 W su di esso, insieme a undici interruttori tattili: per due D-pad e tre pulsanti dell'interfaccia utente. Sono presenti anche connessioni per una batteria LiPo e un mini display OLED opzionale. Una libreria MicroPython semplifica la programmazione.

Un kit alternativo è l'Alpakka 1, basato su un modulo alimentato da RP2040: **inputlabs.io/alpakka**.

Macro Pad

rpimag.co/picomacropad

Un tastierino macro può rivelarsi un'aggiunta molto utile a una configurazione informatica, consentendo di attivare facilmente scorciatoie e sequenze di tasti personalizzate con la semplice pressione di un tasto.

Esistono kit di tastierini macro basati su RP2040/Pico, prodotti da aziende come Pimoroni e Adafruit. In alternativa, è possibile costruirne uno personalizzato, come questo esempio a nove tasti realizzato con un Pico, un Arduino Nano R3, switch Gateron e copritasti. Programmato in CircuitPython, permette di registrare macro da attivare con ogni tasto.

In alternativa, è possibile costruire una tastiera completa basata su Pico, ad esempio con il kit pi40: **rpimag.co/pi40**.

Indossabili

Progetti Pico che puoi indossare

Cyber Glasses

rpimag.co/cyberglasses

Le dimensioni ridotte e il basso consumo energetico di Pico lo rendono ideale per progetti indossabili, come l'aggiunta di illuminazione a LED a costumi da cosplay o cappelli, come il famoso Raspberry Beret (**rpimag.co/picoberet**). È stato utilizzato anche per badge interattivi per conferenze, come la gamma Badgerware di Pimoroni (**rpimag.co/badgerware**).

Kevin McAleer, invece, ha scelto di impreziosire gli occhiali Cyber Glasses, modificabili. Pico è montato su un'asta degli occhiali stampati in 3D, insieme a un servomotore che muove un anello NeoPixel simile a un monocolo davanti all'occhio destro di chi li indossa.

Pip-Boy 2040

rpimag.co/pipboy2040

Oltre alla linea di prodotti ufficiali Pico, esistono numerose schede di terze parti che utilizzano gli stessi chip Raspberry Pi RP2040 e RP2350. Questo progetto di Adafruit, ad esempio, utilizza la scheda Feather 2040 dell'azienda. Ispirato al videogioco *Fallout*, è un accessorio da polso pronto per l'apocalisse! È dotato di un display IPS TFT rettangolare arrotondato in una custodia stampata in 3D. Con il software dimostrativo, scritto in CircuitPython, è possibile passare da una schermata grafica all'altra utilizzando i pulsanti direzionali e muovere il cursore con il mini joystick. Con un po' di fantasia, si potrebbero facilmente creare programmi più interattivi.

Illuminazione a LED

Fai luce con dei LED

Luci Natalizie

rpimag.co/picoxmaslights

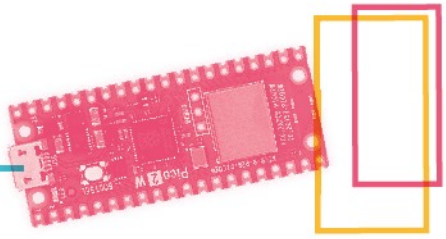
Le stringhe o le strisce di LED indirizzabili individualmente sono perfette per creare effetti multicolore originali. Il nostro esperto di illuminazione a LED, Ben Everard, ha recentemente pubblicato una guida in due parti sul controllo delle luci natalizie con Pico, a partire dal numero 159 (**rpimag.co/159**). Collegare una striscia NeoPixel (nota anche come WS2812B) è semplice, bastano tre fili: alimentazione, massa e ingresso dati. Tuttavia, se si dispone di un numero elevato di NeoPixel (più di 30 circa), potrebbe essere necessario alimentare a parte il circuito per illuminarli tutti alla massima luminosità.

Luci Scala

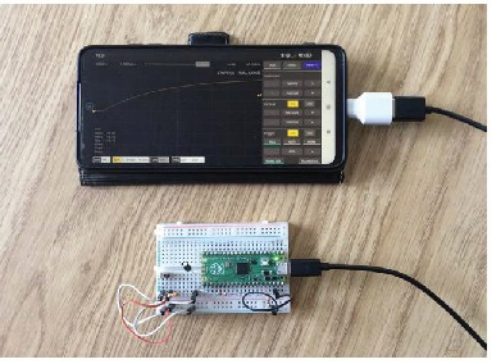
rpimag.co/picostairlights

Aggiungere dei LED a una scala è da tempo un progetto di illuminazione popolare per i dispositivi Raspberry Pi. Il progetto di Craig Robertson utilizza un Pico e una striscia NeoPixel da 30 pixel. Un sensore PIR (infrarossi passivi) rileva una persona che si avvicina e illumina il percorso. Ha anche aggiunto un LDR (resistore dipendente dalla luce) per monitorare il livello di luce ambientale in modo che le luci si accendano solo al buio, non alla luce del giorno. Ha programmato Pico per accendere i LED in sequenza, sfumandoli da un colore al bianco.

Varie



Altri progetti da realizzare con Pico



Scoppy

rpimag.co/scoppy

Abbiamo pubblicato un tutorial su questo oscilloscopio basato su Pico nel numero 134 **rpimag.co/134**. È facile da configurare: basta caricare il firmware personalizzato su Pico e installare l'app dedicata su un telefono Android. Pico può leggere i segnali analogici da un circuito collegato ai suoi pin GPIO ADC e inviarli all'app, che visualizza i risultati su un grafico e offre un'interfaccia di controllo per l'oscilloscopio. È disponibile anche una modalità di analizzatore logico per visualizzare i segnali digitali sui pin GPIO.

Overkill Cyclometer

rpimag.co/cyclometer

Definendolo scherzosamente "un piano per mandare in bancarotta le aziende produttrici di ciclocomputer", Martin Cejp ha costruito questo ciclocomputer fai-da-te alimentato da RP2040 partendo da zero. Un sensore reed magnetico resistente alle intemperie viene utilizzato per contare le rotazioni della ruota e calcolare la velocità del ciclista e la distanza percorsa, visualizzate su un mini schermo OLED. Dopo la prototipazione, Martin ha progettato un PCB personalizzato, sul quale ha montato una scheda Arduino Nano RP2040 Connect, preferita rispetto a una Pico standard per il suo più ampio intervallo di alimentazione in ingresso: da 5 a 18 V, in modo da usare una batteria standard da 9 V. Il tutto è racchiuso in un contenitore stampato in 3D.

Elettronica Con Pico

Con pochi componenti, puoi iniziare il tuo percorso di creazione digitale

COSA SERVE

- Raspberry Pi Pico
- Breadboard
- LED (qualsiasi colore)
- Resistenza da 330Ω
- Pulsante
- Sensore DHT11, esempio **rpimag.co/dht11**
- 3 cavetti jumper maschio-maschio

Prima di realizzare il tuo capolavoro con Pico, ti consigliamo di imparare le basi del controllo dei circuiti elettronici. Inizieremo facendo lampeggiare un LED, poi rileveremo la pressione di un pulsante per accenderlo. Infine, collegheremo un sensore per monitorare temperatura e umidità. Dato che i nostri esempi sono scritti in MicroPython, dovrai averlo installato sul tuo Pico: scaricalo da **rpimag.co/micropython**. Una volta fatto, utilizza un IDE Python sul computer collegato per inserire ed eseguire il codice. Noi utilizziamo Thonny con l'interprete Python impostato su "MicroPython (Raspberry Pi Pico)".

Far lampeggiare un LED

Questo è il "Ciao Mondo" dei progetti di elettronica, in genere la prima cosa che farai. Collega il circuito su una breadboard come mostrato in **Figura 1**. Nota che il piedino più lungo, positivo del LED (l'anodo) è collegato, tramite una resistenza da 330Ω, al pin GPIO 15 di Pico; il piedino più corto (catodo) è collegato a un pin GND (massa). Il LED non si accenderà finché non porteremo il pin GPIO 15 a livello alto, nel nostro programma, per alimentare il circuito.

Immetti il codice da **Blink.py**. All'inizio, importiamo il metodo `Pin` dal modulo `machine` di MicroPython, che consente di leggere e controllare i pin GPIO di Pico. Importiamo anche il metodo `sleep` dal modulo `time` per aggiungere un ritardo tra i lampeggi.

Assegniamo poi il LED al pin GPIO 15, impostandolo come uscita. Infine, in un ciclo `while True:` che viene eseguito ripetutamente, accendiamo e spegniamo il LED, con un ritardo di un secondo tra ogni cambiamento.

Premi il pulsante

Oltre a controllare dispositivi di output come i LED, Pico può leggere i segnali da dispositivi di input come un pulsante. Aggiungeremo un pulsante al nostro circuito esistente, collegandone un lato al pin GPIO 16 di Pico e l'altro al pin di alimentazione da 3,3 V, come mostrato in **Figura 2**. Premendo il pulsante, il circuito si chiuderà.

Pico può leggere i segnali provenienti da dispositivi di input come un pulsante.

Inserisci il codice dal listato **Button.py**. Come prima, importiamo il metodo `Pin` dal modulo `machine` di MicroPython, insieme al metodo `sleep` dal modulo `time`.

Definiamo quindi un oggetto per il pulsante, assegnandolo al pin GPIO 16, ma questa volta come input; poiché il pulsante è collegato a 3,3 V, impostiamo la resistenza del pin su pull-down (vedi "Pull Up o Pull Down" per i dettagli). Infine, utilizziamo un ciclo `while True:` per controllare ripetutamente lo stato del pulsante; se è premuto, attiviamo il LED come prima.

PULL UP O PULL DOWN

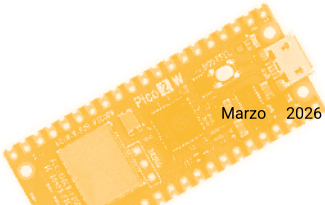
I pin GPIO di Pico hanno resistenze integrate che si possono configurare come pull-down o pull-up. Se si utilizza la pull-down, come nel nostro esempio, il pulsante deve essere collegato a 3,3 V e, quando non premuto, la lettura dell'ingresso è 0. Se si utilizza il pull-up, il pulsante deve essere collegato a GND e, quando non premuto, la lettura dell'ingresso sarà 1.

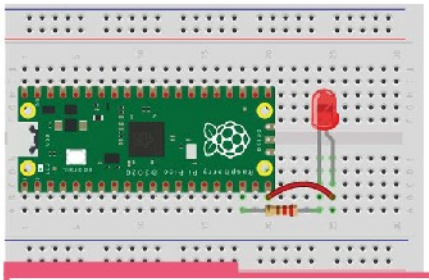
Leggere un sensre

Un altro dispositivo di input è un sensore. Ne esistono molti tipi diversi, con uscite digitali e/o analogiche. Pico ha alcuni ingressi analogici, ma in questo caso stiamo leggendo un sensore digitale: un DHT11 che misura la temperatura e la pressione atmosferica.

Per iniziare un nuovo circuito, collega il sensore a Pico come mostrato in **Figura 3**: i collegamenti sono 3V3 (su Pico) a VCC (sul DHT11), GND a GND e GPIO 14 a DOUT.

Inserisci il codice dal listato **Sensor.py**. In alto, importiamo il modulo `dht11` per il sensore, insieme al metodo `Pin` da `machine`. Definiamo un oggetto per il sensore, assegnandolo al pin GPIO 14 come input. Quindi effettuiamo una misurazione, filtrandola per valori di temperatura e umidità. Infine, stampiamo i risultati nella Shell.

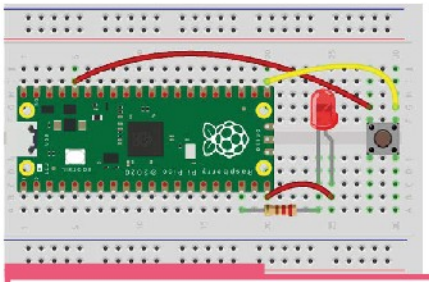




Blink.py

> Linguaggio: **MicroPython**

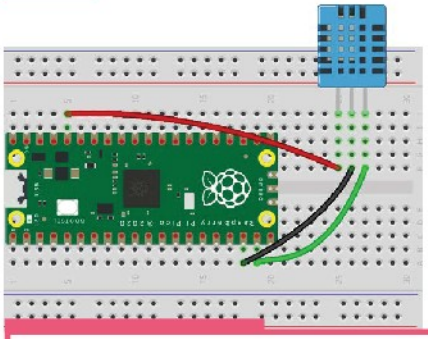
```
001.  from machine import Pin
002.  from time import sleep
003.
004.  led = Pin(15, Pin.OUT)
005.
006.  while True:
007.      led.value(1) # accendi il LED
008.      sleep(1)     # attendi 1 secondo
009.      led.value(0) # spegni il LED
010.      sleep(1)     # attendi 1 secondo
```



Button.py

> Linguaggio: **MicroPython**

```
001.  from machine import Pin
002.  from time import sleep
003.
004.  led = Pin(15, Pin.OUT)
005.  button = Pin(16, Pin.IN, Pin.PULL_DOWN)
006.
007.  while True:
008.      if button.value() == 1:
009.          led.value(1)
010.      led.value(0)
```



Sensor.py

> Linguaggio: **MicroPython**

```
001.  import dht
002.  from machine import Pin
003.
004.  sensor = dht.DHT11(Pin(14))
005.
006.  sensor.measure()
007.  temp = sensor.temperature()
008.  hum = sensor.humidity()
009.
010.  readings = ("Temperatura: {}°C
      Umidità: {}% ".format(temp, hum))
011.  print(readings)
```

Ulteriori letture

Scopri di più su Pico e sull'elettronica consultando queste risorse:

Get Started with MicroPython on Raspberry Pi Pico (rpimag.co/picobook) – Una volta terminato il nostro libro, saprai come creare i tuoi dispositivi elettronici programmabili

Code Club Projects (rpimag.co/ccpicoprojects) – Code Club offre una vasta gamma di progetti divertenti per aiutarti a iniziare con Pico e imparare a programmare.

Pico Documentation (rpimag.co/picodoc) – La documentazione ufficiale online contiene tutti i dettagli, incluso un pratico schema dei pin del Pico.

Wokwi Pico Simulator (wokwi.com/pi-pico) – Questo pratico sito ti permette di simulare circuiti Pico ed eseguire codice per testarlo.

▶ Il nostro libro ti aiuterà a iniziare a realizzare progetti Pico